Amendments to the Claims:

Please amend claims 1, 3, 5, 13-14, 23, 34, and 36 as follows:

1. (currently amended): A method of compiling a high level language to map a plurality of tasks and a plurality of data onto a very long instruction word configurable multiple processor, distributed memory hardware architecture, the method comprising:

a) describing a task-level network of behaviors that defines an embedded system in the very long instruction word configurable multiple processor, distributed memory hardware architecture, each of the task-level network of behaviors being related to each other through control and data flow;

b) predicting a schedule of tasks for the task-level network of behaviors;

c) allocating the plurality of tasks to at least ~~one~~ two processors and allocating the plurality of data to at least ~~one~~ two distributed ~~memory~~ memories in the very long instruction word configurable multiple processor, distributed memory hardware architecture in response to the predicted schedule of tasks; and

d) generating machine executable code with the allocated plurality of tasks and allocated plurality of data for the very long instruction word configurable multiple processor, distributed memory hardware architecture that enables parallel execution of very long instruction word operations.

2. (original): The method of claim 1 wherein the predicting the schedule of tasks comprises minimizing execution time of the plurality of tasks.

3. (currently amended): The method of claim 2 wherein the predicting the schedule of tasks comprises minimizing the schedule of tasks by allocating data to at least ~~one~~ two distributed memory in the configurable multiple processor, distributed memory hardware architecture in order to minimize data transfers.

4. (previously presented): The method of claim 1 wherein the predicting the schedule of tasks comprises maximizing parallel execution of the plurality of tasks on at least two processors in the configurable multiple processor, distributed memory hardware architecture.

5. (currently amended): The method of claim 1 wherein the allocating the plurality of tasks comprises allocating tasks to at least ~~one~~ two processor in the configurable multiple processor, distributed memory hardware architecture, which has optimal processor resources for the tasks.

6. (previously presented): The method of claim 1 wherein the predicting the schedule of tasks comprises using a resource-based model of the configurable multiple processor, distributed memory hardware architecture to predict the schedule of tasks.

7. (original): The method of claim 1 wherein the predicting the schedule of tasks comprises using an interval graph and an execution time model of the task-level network of behaviors to predict the schedule of tasks.

8. (original): The method of claim 1 wherein the allocating the plurality of tasks and data comprises an iterative allocation process.

9. (original): The method of claim 8 wherein the iterative allocation process comprises

using a demand-driven and constraint-based objective function.

10. (original): The method of claim 1 wherein the describing a task-level network of behaviors comprises describing a task-level network of behaviors in a high-level programming language.

11. (original): The method of claim 10 wherein the describing the task-level network of behaviors in the high-level programming language comprises parsing the high-level programming language into an intermediate form.

12. Cancelled.

13. (currently amended): The method of claim 1 wherein the allocating the plurality of data to the at least ~~one~~ two distributed ~~memory~~ memories in the configurable multiple processor, distributed memory hardware architecture comprises allocating data to shared memories.

14. (currently amended): The method of claim 1 wherein the allocating the plurality of data to the at least ~~one~~ two distributed ~~memory~~ memories in the configurable multiple processor, distributed memory hardware architecture comprises allocating data to private memories.

15. Cancelled.

16. Cancelled.

17. Cancelled.

18. Cancelled.

19. Cancelled.

20. Cancelled.

21. Cancelled.

22. Cancelled.

23. (currently amended): A method of compiling a high level language to a schedule of tasks in a very long instruction word configurable multiple processor, distributed memory architecture, the method comprising:

a) generating the schedule of tasks based at least in part on a task-level network of behaviors that defines an embedded system;

b) calculating a demand function based at least in part on a constraint related to at least one of a plurality of tasks in the schedule of tasks;

c) allocating a task having highest priority to a processor having least cost according to the demand function;

d) generating machine executable code with the allocated task for the very long instruction word configurable multiple processor, distributed memory hardware architecture that enables parallel execution of very long instruction word operations.

24. (original): The method of claim 23 further comprising allocating a data block to a memory in the distributed memory.

25. (original): The method of claim 23 wherein the demand function is calculated based at least in part on the task-level network of behaviors.

26. (original): The method of claim 23 wherein the demand function is calculated based at least in part on an impact on the schedule of tasks.

27. (original): The method of claim 23 wherein the demand function is calculated based at least in part on an impact on data movement.

28. (original): The method of claim 23 wherein the demand function is calculated based at least in part on prior allocation decisions.

29. (previously presented): The method of claim 23 wherein the cost is defined as a least negative impact on at least one performance factor.

30. (original): The method of claim 29 wherein the at least one performance factor comprises the schedule of tasks.

31. (original): The method of claim 29 wherein the at least one performance factor comprises data movement.

32. (original): The method of claim 23 further comprising allocating a task having next highest priority to a processor having next least cost according to the demand function.

33. (original): The method of claim 23 further comprising recalculating the demand function in response to each task in the plurality of tasks being allocated to a processor.

34. (currently amended): A compiler for mapping a plurality of tasks and data onto a very long instruction word configurable multiple processor, distributed memory architecture,

the compiler comprising:

a) means for describing a task-level network of behaviors that defines an embedded system, each of the task-level network of behaviors being interrelated through control and data flow dependencies;

b) means for predicting a schedule of tasks for the task-level network; and

c) means for allocating the plurality of tasks and data to at least ~~one~~ two processors and to at least ~~one~~ two distributed ~~memory~~ memories in the very long instruction word configurable multiple processor, distributed memory architecture, respectively, in response to the predicted schedule of tasks to enable parallel execution of very long instruction word operations.

35. (previously presented): The compiler of claim 34 further comprising a means for producing machine executable code for the configurable multiple processor, distributed memory hardware architecture based at least in part on the means for allocating the plurality of tasks and data.

36. (currently amended): The compiler of claim 34 wherein at least ~~one~~ two processor in the configurable multiple processor, distributed memory architecture communicates using at least ~~one~~ two distributed ~~memory~~ memories in the configurable multiple processor, distributed memory architecture.